How to Scale Jenkins for CD with Microsoft Azure

CloudBees Inc
Brian Dawson and Tyler Croy
@brianvdawson @agentdero

Microsoft
Thiago Almeida and Oguz Pastirmaci
@nzthiago @OguzPastirmaci
Game plan

- CD is what again??
- The State of CD
- BS #!*@! to AS :^)
- Azure Container Service
- This is your Jenkins on Azure
Poll (1/2)

- How big is your Development/IT organization?
  a) 1 - 50 ppl
  b) 50 - 200 ppl
  c) 200 - 500 ppl
  d) Other or "Do you know anybody hiring?"

- What is your role?
  a) Developer
  b) Operations
  c) QA
  d) All of the above
  e) Other
Poll (2/2)

• By show of hands do you practice?
  a) Continuous Integration
  b) Continuous Delivery
  c) DevOps
  d) None of the above or GDD (Get ‘er Done Development)

• At what scale (tooling and practice)?
  a) Single Team
  b) Multiple Teams
  c) Organization-wide (managed services)
  d) Cloud-Scale (internal or external PaaS)
CD is what again??
Fundamentally it’s about...

Delivering Better Software Faster!

- Concept-to-Customer
- High Quality
  - Performance
  - Feature
  - Function
- High Value
- Rapidly Eliminating Waste
- Market Innovation

Microsoft Azure

Jenkins
Continuous delivery (CD) is...

• the practice of automatically and continuously building, testing and deploying software to ensure it **CAN** be released at any time.

• an extension of continuous integration (CI) and enables DevOps

• NOT Continuous Deployment
How Do You Deliver Better Software Faster?

Dev ➔ Prod

Rapidly
Repeatedly
Reliably
The State of Continuous Delivery

• CD offers significant technical and business benefits
  - Deliver Faster - ship to production several times per week or per day
  - Demonstrably higher quality
  - Considerably lower production incidents and near-zero Sev 1’s
  - No War Rooms, imagine the harmony
  - Measurably happier team members

...and more
The State of Continuous Delivery

- CD is being rapidly adopted across teams and organizations

65% of respondents have started down the path to implementation of continuous delivery with 37% adopting it on some projects, and 28% on all projects.

- Perforce “Continuous Delivery: The New Normal for Software Development”
The State of Continuous Delivery

- CD is being rapidly adopted across teams and organizations

65% of respondents have started down the path to implementation of continuous delivery with 37% adopting it on some projects, and 28% on all projects.

- Perforce “Continuous Delivery: The New Normal for Software Development”
Jenkins, CI and CD tooling are often implemented at team level and administered by developers.

Yay! I deployed Jenkins and everybody is using it!

Damn! Now I own it!#@!*

The State of Continuous Delivery
The State of Continuous Delivery

- Jenkins, CI and CD have become mission critical

Would you consider Jenkins to be mission-critical to your development process?

- 83.12% (596)
- 16.88% (119)

Yes

No
The State of Continuous Delivery

- Jenkins is the leading CD solution

What types of tasks do you use Jenkins for?

- Build
- Test
- Code quality analysis
- Batch tasks
- Release
- Operation
- Deployment
- Other (please specify)

70% Jenkins
Most used CI server in the industry
This results in....

- Developers acting as administrators
- Potential lack of security and/or exposure of IP
- Difficulty in sharing best practices across teams
- Absence of reliable infrastructure to maintain continuity and productivity
- Inability to quickly scale as adoption or demand grows
BS #!@@@ to AS :^)
Benefits of Scaling Continuous Delivery

- Able to recognize the benefits across the entire organization
- Enable teams to rapidly on-board and adopt
- Remove team-level administration to increase reliability and productivity, and governance
- Provide cross-team visibility
- Share best practices
Before Scale and After Scale

Your changes had few chances of shipping the next day

Your changes have multiple opportunities to ship each day
Before Scale and After Scale

Local provisioning or limited access to cloud environments

Elastic cloud platform
Before Scale and After Scale

Inability for teams to rapidly and consistently provision environments

Teams can provision and repeat prod-like environments as needed
Before Scale and After Scale

- Centralized and bottleneck infrastructure team
- Infrastructure admins that support team-level independence (RBAC, etc)
Before Scale and After Scale

One person/team responsible for knowledge

Visibility and Analytics across org and knowledge sharing
Before Scale and After Scale

Downtime, unavailable infrastructure

High-availability and fault tolerance, downtime is deadly
Before Scale and After Scale

Weeks to months to onboard new teams into a CD environment

Instant provisioning of a CD infrastructure
- Azure and Azure Container Service
- Jenkins in the Cloud